Decomposition and Metaoptimization of Mutation Operator in Differential Evolution

Karol Opara[1] and Jarosław Arabas[2]

[1] Systems Research Institute, Polish Academy of Sciences
[2] Institute of Electronic Systems, Warsaw University of Technology
karol.opara@ibspan.waw.pl,jarabas@elka.pw.edu.pl

Abstract. Metaoptimization is a way of tuning parameters of an optimization algorithm with use of a higher-level optimizer. In this paper it is applied to the problem of choosing among possible mutation range adaptation schemes in Differential Evolution (DE). We consider a new version of DE, called DE/rand/∞. In this algorithm, differential mutation is replaced by a Gaussian one, where the covariance matrix is determined from the contents of the current population. We exploit this property to separate the adaption of search directions from the adaptation of mutation range. The former is characterized by a norm of the covariance matrix while the latter can be expressed as a normed covariance matrix multiplied by the scaling factor. Such separation allows us to introduce a few schemes of direct, explicit control of the mutation range and to compare them with the basic, implicit scheme present in DE/rand/∞. To ensure fair comparisons all versions of DE/rand/∞ are first metaoptimized and then assessed on the CEC'05 benchmark.

Keywords: differential evolution, metaoptimization, adaptation of mutation

1 Introduction

Tuning an optimization algorithm consists in finding values of its parameters that ensure its maximal performance. This can be seen as an optimization problem in the space of parameters. The process of applying an optimizer to tune parameter values of another optimization method is called metaoptimization and has been used since at least 1980's [2]. In this paper it is applied to the problem of choosing the most effective mutation adaptation scheme in a novel modification of differential evolution algorithm DE/rand/∞ which we introduced in [7]. Each variant of the algorithm is tested on a subset of CEC'05 benchmark functions [11] in order to choose the best-performing one. Reliable comparison of variants of algorithm requires tuning parameters for each of them, which can be achieved by means of metaoptimization. Maximizing performance for a set of test functions can be a noisy multiobjective optimization task with both discrete and continuous variables, which are often subject to constraints. For these reasons, metaoptimization is a non-trivial and very computationally expensive task.

This paper aims at summarizing experiences of choosing mutation operator for DE/rand/∞ with use of metaoptimization. In classical DE, both range and direction of mutation are implicitly adopted through the use of difference vectors. Introduction of DE/rand/∞ algorithm (section 2) allows to explicitly control mutation range without hindering the property of adaptation of search directions. In this paper a few explicit methods of controlling mutation range are defined and compared with the original, implicit adaptation scheme. Each of the resulting DE/rand/∞ variants becomes then subject to a metaoptimization procedure discussed in section 3. The paper is concluded with a discussion of the metaoptimized parameter values.

2 From DE/rand/1 to DE/rand/∞

DE/rand/1. Differential evolution (DE) is a simple and effective continuous stochastic optimizer [9], whose outline is presented as Algorithm 1. The fitness function is denoted by f, $\mathbf{P}^t$ is the population in generation t and $\mathbf{P}^t_i$ denotes the i-th individual. The algorithm takes three parameters: population size NP, crossover probability CR and scaling factor F which is used for mutation.

For every individual $\mathbf{P}^t_i$, another individual $\mathbf{P}^t_{i_1}$ is randomly selected. A mutant $\mathbf{u}_i$ is created by adding a scaled difference between two other randomly picked individuals $\mathbf{P}^t_{i_2}$ and $\mathbf{P}^t_{i_3}$ to individual $\mathbf{P}^t_{i_1}$.

$$\mathbf{u}_i \leftarrow \mathbf{P}^t_{i_1} + F \cdot (\mathbf{P}^t_{i_2} - \mathbf{P}^t_{i_3}) \tag{1}$$

The mutant $\mathbf{u}_i$ is then crossed-over with individual $\mathbf{P}^t_i$. Differential mutation is directly dependent on the spread of current population through the use of the difference vectors $F \cdot (\mathbf{P}^t_{i_2} - \mathbf{P}^t_{i_3})$. This leads to an implicit adaptation of range and direction of differential mutation. In our opinion, this adaptation mechanism coupled with the greedy local selection scheme are the main reasons for high performance of DE.

DE method, which uses mutation operator defined in equation (1) is called DE/rand/1, since there is only one difference vector and the index i_1 is chosen randomly with uniform distribution. Observe that scaled difference vector $F \cdot (\mathbf{P}^t_{i_2} - \mathbf{P}^t_{i_3})$ is a random variable, whose distribution depends on the population contents and can be expressed by means of convolution of distributions [7]. Fig. 1 a) shows a population spread in a two-dimensional space, while Fig. 1 b) presents the corresponding difference vector distribution. This distribution is symmetric with respect to origin, has zero mean and its covariance matrix is proportional to the covariance matrix of vectors in the current population.

$$\mathrm{cov}\left(F \cdot (\mathbf{P}^t_{i_2} - \mathbf{P}^t_{i_3})\right) = 2F^2 \mathrm{cov}(\mathbf{P}^t) \tag{2}$$

Equation (2) shows that range and direction of differential mutation is implicitly dependent on contents of the current population $\mathbf{P}^t$.

Algorithm 1 Differential Evolution

```
Initialize parameters: $CR$, $F$, and $NP$
Initialize population $\mathbf{P}^0$, $t \leftarrow 0$
while stop condition not met do
    for all $i \in \{1, 2, ..., NP\}$ do
        $\mathbf{u}_i \leftarrow \texttt{mutation}(F; i, \mathbf{P}^t)$
        $\mathbf{o}_i \leftarrow \texttt{crossover}(CR; \mathbf{P}^t_i, \mathbf{u}_i)$
        if $f(\mathbf{o}_i) \leq f(\mathbf{P}^t_i)$ then
            $\mathbf{P}^{t+1}_i \leftarrow \mathbf{o}_i$
        else
            $\mathbf{P}^{t+1}_i \leftarrow \mathbf{P}^t_i$
        end if
    end for
    $t \leftarrow t + 1$
end while
return arg $\min_i f(\mathbf{P}^t_i)$
```

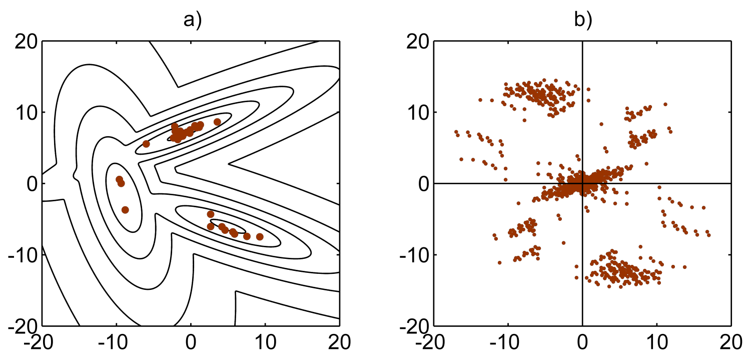


Fig. 1. Population scattered in the search space a), corresponding difference vector distribution b)

DE/rand/∞ [7]. Differential mutation may be generalized by using k difference vectors [9], which is denoted by DE/rand/k ($k = 1$ and $k = 2$ are the most common choices).

$$\mathbf{u}_i \leftarrow \mathbf{P}^t_{i_1} + F \cdot (\mathbf{P}^t_{i_2} - \mathbf{P}^t_{i_3}) + F \cdot (\mathbf{P}^t_{i_4} - \mathbf{P}^t_{i_5}) + ... + F \cdot (\mathbf{P}^t_{i_{2k}} - \mathbf{P}^t_{i_{2k+1}}) \quad (3)$$

Indices i, i_1, i_2, ..., i_{2k+1} are required to be pairwise distinct. If we drop this assumption, then picking each difference vector $F \cdot (\mathbf{P}^t_{j_1} - \mathbf{P}^t_{j_2})$ would be equivalent to realization of a random variable. Its distribution is determined by the current population $\mathbf{P}^t$ and exemplified in Fig. 1. Hence, summing k difference vectors is equivalent to summing k independent, identically distributed random variables with zero mean and covariance matrix given by (2). The covariance matrix of difference vectors for DE/rand/k equals $2kF^2\text{cov}(\mathbf{P}^t)$ which implies that the range of change introduced by the mutation (3) will increase with k. This effect can be eliminated by dividing the sum by $\sqrt{k}$:

$$\mathbf{u}_i \leftarrow \mathbf{P}^t_{i_1} + \frac{F}{\sqrt{k}} \sum_{j=1}^{k} \left(\mathbf{P}^t_{i_{2j}} - \mathbf{P}^t_{i_{2j+1}} \right) \quad (4)$$

On the basis of central limit theorem the distribution of the normed sum of difference vectors 4 weakly converges to the normal distribution with zero mean and the covariance matrix equal to $2F^2\text{cov}(\mathbf{P}^t)$. Consequently, under assumption that $k \to \infty$ one can replace 4 by:

$$\mathbf{u}_i \leftarrow \mathbf{P}^t_{i_1} + \sqrt{2}F \cdot \mathbf{v}_\infty, \text{ where } \mathbf{v}_\infty \sim \mathcal{N}\left(0, \text{cov}(\mathbf{P}^t)\right). \quad (5)$$

Thus, if we drop the assumption that indices i, i_1, ..., i_{2k+1} must be pairwise distinct, we can replace the differential mutation by the Gaussian mutation with zero mean and covariance matrix proportional to the covariance matrix of the current population. Our earlier analyzes [7] show that performance of DE/rand/∞/bin is comparable to DE/rand/1/bin and may be improved by coupling with an exploitative mutation operator DE/best/1.

Decomposition of mutataion Formula (5) can be reformulated as follows:

$$\mathbf{u}_i \leftarrow \mathbf{P}^t_{i_1} + F \cdot \sqrt{2||\text{cov}(\mathbf{P}^t)||} \cdot \mathbf{v}_i, \text{ where } \mathbf{v}_i \sim \mathcal{N}\left(0, \frac{\text{cov}(\mathbf{P}^t)}{||\text{cov}(\mathbf{P}^t)||}\right) \quad (6)$$

Observe that mutation range is decomposed to a product of the scalar factor F and a scalar describing the spread of the current population, which we measure as the covariance matrix norm $\sqrt{||\text{cov}(\mathbf{P}^t)||}$. Vector $\mathbf{v}_i$ describes the direction of differential mutation. Decomposition (6) allows us to separately analyze mutation range and direction in DE/rand/∞.

Explicit control of mutation range. In this study we were interested in analysis of the implicit adaptation mechanism in DE. Decomposition (6) shows that mutation range in DE/rand/∞ is proportional to $\sqrt{||\text{cov}(\mathbf{P}^t)||}$. A natural question arises, how does it compare to other possible ways of controlling mutation range? To answer this question we modified the scheme (6) by substituting the product of the scaling factor and the root of covariance matrix norm with a function dependent on the generation index.

$$\mathbf{u}_i \leftarrow \mathbf{P}^t_{i_1} + \sqrt{2}F(t)\overline{s} \cdot \mathbf{v}_i, \text{ where } \mathbf{v}_i \sim \mathcal{N}\left(0, \frac{\text{cov}(\mathbf{P}^t)}{||\text{cov}(\mathbf{P}^t)||}\right) \tag{7}$$

This provides explicit control over the mutation range while preserving adaptation of search directions. The constant $\overline{s}$ adjusts the scaling factor to the size of a feasible set. In this paper, each fitness function f_i, $i \in \{1, 2, ..., 14\}$ was constrained to a hypercube $[l_i, u_i]^n$ and the value of $\overline{s}$ for i-th problem was defined as $u_i - l_i$. Introduction of the $\overline{s}$ term allows to reinterpret the scaling factor F. Intuitively speaking, $F \approx 1$ means that the mutation range is approximately the same as the size of the feasible set, $F \approx 0.1$ mean that it is 10 times smaller etc. We defined three variants of dynamic control of mutation range (7), namely using a constant value, decreasing it linearly and exponentially—see Table 1. In addition we also considered a noisy version of the `const` strategy and methods of periodic change along sawtooth `saw` and sine `sin`, but they performed consistently worse.

3 Metaoptimization procedure

Choice of test problems. To our knowledge, there are no theoretical clues about optimal mutation range adaptation. Performance of different methods of mutation adaptation was hence measured on a benchmark of ten-dimensional test problems introduced at the CEC'05 [11] conference. The CEC'05 benchmark contains 5 unimodal functions, 7 basic multimodal ones as well as two multimodal complex functions. Apart from that, there are 11 hybrid functions, each of whom is created as a weighted sum of 10 basic ones. In general, hybrid problems proved to be too complicated to be solved [3]. Therefore, we decided to limit metaoptimization to the first 14 problems only. There are also newer and more elaborated global optimization benchmarks, in particular BBOB [5]. We decided to use CEC'05 mainly because it defines stopping condition based on maximal number of function evaluations, which is convenient in case of dynamic control of mutation range.

In this study an implementation of CMA-ES [4] was used as a metaoptimizer. All investigated mutation range adaptation variants were started with the same seed values of the random number generator.

Table 1. Mutation range adaptation schemes

	Parameters	Adaptation scheme
`implicit`	$NP \in \mathbb{N}$, $F_0 \in \mathbb{R}$	$F = F_0$, no norming; mutation according to (5)
`const`	$NP \in \mathbb{N}$, $F_0 \in \mathbb{R}$	$F(t) = F_0$
`lin`	$NP \in \mathbb{N}$, $F_0 \in \mathbb{R}$	$F(t) = F_0 \frac{t_{max}-t}{t_{max}}$
`exp`	$NP \in \mathbb{N}$, $F_0, F_1 \in \mathbb{R}$	$F(t) = F_0 \left(\frac{F_1}{F_0}\right)^{t/t_{max}}$

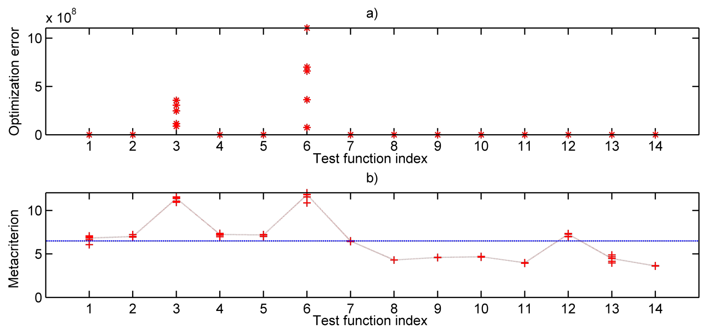


Fig. 2. Derivation of metacriterion, a) raw error values plotted for 5 runs for each of 14 test problems; b) logarithms of the errors (8), medians connected with a dotted line, metacriterion shown by the horizontal line

Metacriterion. Performance for each test problem was assessed on the basis of the final optimization error after 10^5 function evaluations. The results for $N = 14$ investigated problems were aggregated to form a single-objective metaoptimization criterion (metacriterion). Due to a random initialization and stochastic nature of DE, final error values were nondeterministic. Therefore, each algorithm was independently restarted k times. The median value of final optimization errors for i-th test problem is denoted by ϵ^i. The metacriterion f_m is defined as

$$f_m = \frac{1}{N} \sum_{i=1}^{N} \left(m + \log_{10}\left(10^{-m} + \epsilon^i\right)\right), \tag{8}$$

where $m = 3$ is a parameter ensuring that the metacriterion takes nonnegative values. It also provides a lower bound on the required error level 10^{-m}. We used the logarithmic transformation to reduce a risk that a single problem with the highest error would dominate all other problems within the benchmark set. For instance, in Fig. 2 a), error values for problems number 3 and 6 are of several orders of magnitude greater than all others. Without the logarithmic transformation, metaoptimizer would fit parameters to increase performance for these two problems only.

Choosing the “best” mutation range adaptation scheme basing on benchmark results is justified when the same criterion is used in the metaoptimization and in the evaluation of the final (metaoptimized) benchmark results. There is however a possibility that parameters would be overfitted to the benchmark. Yet, currently available benchmarks are still quite difficult to solve, even for the state-of-the-art optimizers [3, 5], so overfitting is arguably not a real threat.

Metalandscapes. A metalandscape graph is the plot of the metacriterion values versus its parameters. Fig. 3 a) shows the metalandscape of DE/rand/∞/none (when $CR = 1$) with the implicit mutation (5). The scaling factor takes values

Table 2. Parameter values obtained through metaoptimization

Mutation	Metaoptimized paremeter values
`implicit`	$NP = 11.5 \cdot n$, $F_0 = 0.54$
`const`	$NP = 4.5 \cdot n$, $F_0 = 0.064$
`lin`	$NP = 5.2 \cdot n$, $F_0 = 0.14$
`exp`	$NP = 9.2 \cdot n$, $F_0 = 350$, $F_1 = 8.3 \cdot 10^{-9}$

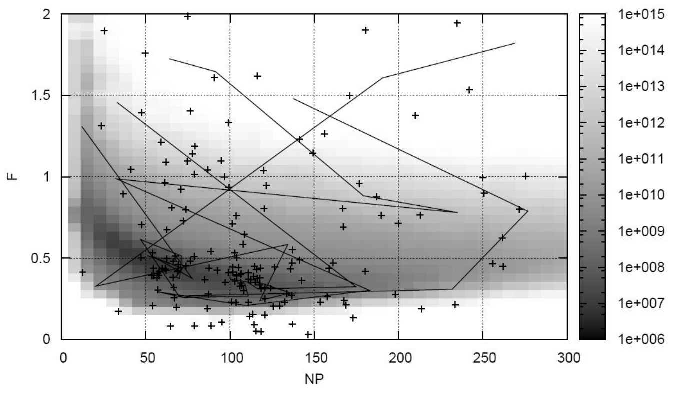
$F_0 \in \{0.4, 0.5, ..., 1.2\}$ while the quotient (NP/n) of population size NP and search space dimension $n = 10$ takes values $(NP/n) \in \{2, 3, 5, 10, 20, 50, 100\}$. Fig. 3 b) presents analogous results published in [8] which were obtained for DE/rand/1/bin for other set of test problems in $n = 30$ dimensions and for a metacriterion defined as the weighted sum of final error values. Nevertheless, the bent shapes of metalandscape are similar in both cases. This may suggest that the metaoptimization method presented here yields robust results and that both algorithms DE/rand/1 and DE/rand/∞ reveal a similar pattern of parameters' influence on the performance on benchmarks. Additionally, conversion of the linear scale to the logarithmic one, as in figures 3 b) and c), seems to improve the conditioning of the metacriterion making it "easier to solve" and "more convex". Consequently, the metaoptimization procedure was applied to logarithms of parameters rather than their raw values.

Interpretation of results. Table 2 contains parameter values obtained during the metaoptimization. Their analysis may give some clues to further the DE/rand/∞ algorithm. First of all, only the implicit and exponential schemes yielded significantly better performance than any other method. Results of the Dunn's and Holm's tests adjusted for multiple comparisons are summarized in Table 3, where significance larger than $1 - \alpha = 0.95$ is denoted by $+$ and lack of it by $\cdot$.

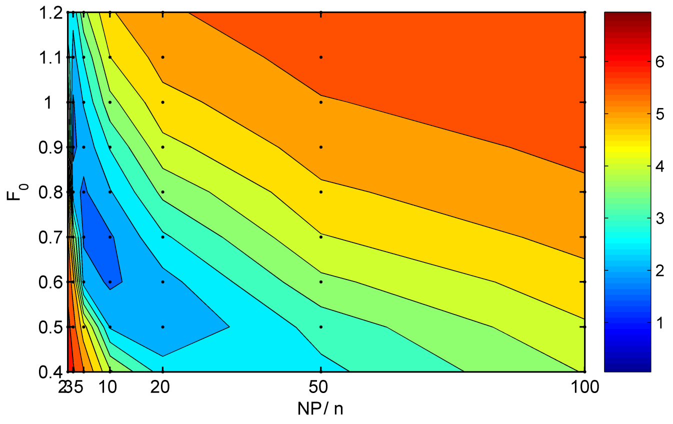
In is noteworthy that for both winning methods population size was of the order $10 \cdot n$ which agrees well with the suggestions for tuning DE given e.g. by Price and Storn [9]. Closer look at the exponential method reveals that the initial mutation range value is huge ($F_0 = 350$) and that it decreases to a very low level ($F_1 = 8.3 \cdot 10^{-9}$). Consequently, for one third of the optimization time, applying differential mutation results in random sampling, since it is nearly entirely guided by a constraint handling method. High performance of the exponential scheme suggests that extending the initialization phase by a period of random sampling compiled with the greedy parent-offspring selection may improve the overall performance of DE/rand/∞.

Table 3. Statistical superiority tests: Dunn's—left hand side, Holm's—right hand side

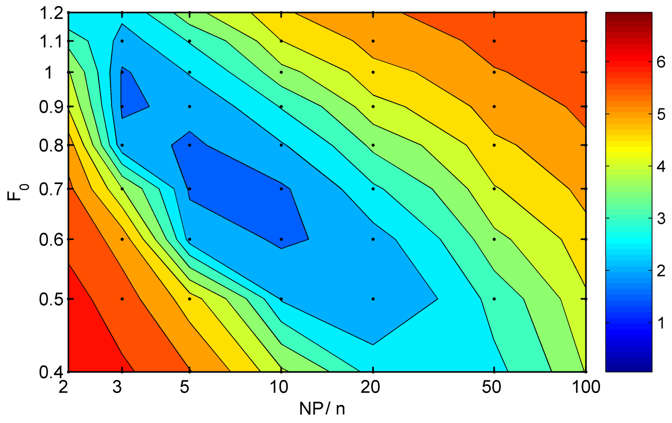
	`implicit`	`exp`	`lin`	`saw`	`sin`	`rand`	`const`
`implicit`		$\cdot\ \cdot$	$\cdot\ \cdot$	$+\ +$	$+\ +$	$+\ +$	$+\ +$
`exp`	$\cdot\ \cdot$		$\cdot\ \cdot$	$+\ \cdot$	$+\ \cdot$	$+\ \cdot$	$+\ +$



(a) DE/rand/1/bin in 30 dimensions [8], linear scale



(b) DE/rand/∞/none in 10 dimensions, linear scale



(c) DE/rand/∞/none in 10 dimensions, log scale

Fig. 3. Metalandscapes for DE/rand/1/bin in linear scale (a) and DE/rand/∞ in linear (b) and logarithmic (c) scales

4 Discussion

Analysis of mutation in DE. Parameter setting in DE has been subject of considerable study [9], [8], as well as various modifications of mutation operators, some of which are surveyed in [6]. Mutation range adaptation in DE was also enhanced by introducing self-adaptation of parameters in jDE [1] and self-adaptation of both parameters and the mutation strategy in SADE [10]. In general, research on differential mutation concentrates on choosing the value of a scale factor F or appropriate variants of mutation operators. The implicit dependence of mutation range and direction on the spread of current population is usually kept as an effective adaptation scheme. This paper provides decomposition (6) of mutation operator in DE/rand/∞. Adaptation of mutation range and adaptation of search directions can be therefore analyzed (or controlled) separately, which provides new opportunities for improving DE. Similar decompositions can be derived for other mutation operators, such as DE/rand/1 or DE/best/1. In such cases the distribution of a random vector $\mathbf{v}_i$ is not normal but depends on the current population in a manner shown in Fig. 1.

Concluding remarks. In this paper we reported an ongoing research on adaptation in DE. We used a metaoptimization approach to consider possible alternative methods to vary the mutation range. From the obtained results it appears that the implicit adaptation method is indeed very effective. It appears however that performance of DE could be improved by prolonging the population initialization phase with a period of sampling with the uniform distribution from the feasible area together with the local selection of results. Further research concentrates on finding functions which simultaneously control mutation range and approximate the implicit adaptation scheme. In this way we hope to explicitly model and analyze the process of mutation range adaptation in DE.

This study was partially supported by research fellowship within “Information technologies: research and their interdisciplinary applications” agreement number POKL.04.01.01-00-051/10-00

References

1. J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *Evolutionary Computation, IEEE Transactions on*, 10(6):646–657, 2006.
2. J.J. Grefenstette. Optimization of control parameters for genetic algorithms. *Systems, Man and Cybernetics, IEEE Transactions on*, 16(1):122 –128, jan. 1986.
3. N. Hansen. Compilation of results on the 2005 CEC benchmark function set, 2006.
4. N. Hansen. The CMA evolution strategy webpage, November 2009.
5. N. Hansen, A. Auger, R. Ros, S. Finck, and P. Posik. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009, 2010.
6. F. Neri and V. Tirronen. Recent advances in differential evolution: a survey and experimental analysis. *Artificical Intelligence Reviews*, 33(1-2):61–106, 2010.

7. K. Opara and J. Arabas. Differential mutation based on population covariance matrix. In R. Schaefer, editor, *Parallel Problem Solving from Nature PPSN XI, part I*, volume 6238 of *LNCS*, pages 114–123. Springer, 2010.
8. M. Pedersen. *Tuning & Simplifying Heuristical Optimization.* PhD thesis, University of Southampton, 2010.
9. K. Price, R. Storn, and J. Lampien. *Differential evolution. A practical approach to global optimization.* Springer, 2005.
10. A. Qin, V. Huang, and P. Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Transaction on Evolutionary Computation*, 13(2):398–417, 2009.
11. P. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical report., 2005.